Authentication

[IdentityCore 2](#_Toc52125002)

[1 Introduction 2](#_Toc52125003)

[JWT Authentication 3](#_Toc52125004)

[1 General 3](#_Toc52125005)

[2 JWT base setup 3](#_Toc52125006)

[3 Token generation 3](#_Toc52125007)

[4 Token consuming 4](#_Toc52125008)

# IdentityCore

## 1 Introduction

Links:

1. GitHub: <https://github.com/dotnet/aspnetcore/tree/master/src/Identity>
2. MSDN: <https://docs.microsoft.com/en-us/aspnet/core/security/authentication/identity?view=aspnetcore-3.1&tabs=visual-studio>
3. Metanit: <https://metanit.com/sharp/aspnet5/16.1.php>

# JWT Authentication

## 1 General

Links:

1. Metanit: <https://metanit.com/sharp/aspnet5/23.7.php>

Assemblies:

1. Microsoft.AspNetCore.Authentication.JwtBearer
2. System.IdentityModel.Tokens.Jwt

JWT token – is an encoded string which may contain special user info (identity info or claims: login, role and etc).

Shortly about auth process:

1. Generate token using user claims, current datetime + additional info and save in response
2. Store token on FE side
3. Receive token from FE then validate it (authentication + authorization) and extract claims info.

## 2 JWT base setup

To use authentication via tokens, need to update Startup. ConfigureServices with: services.AddAuthentication(JwtBearerDefaults.AuthenticationScheme)

.AddJwtBearer(options => //setup code);

## 3 Token generation

1. Create user claims

var claims = new List<Claim>

{

new Claim(ClaimsIdentity.DefaultNameClaimType, person.Login),

new Claim(ClaimsIdentity.DefaultRoleClaimType, person.Role)

};

ClaimsIdentity claimsIdentity =

new ClaimsIdentity(claims, "Token", ClaimsIdentity.DefaultNameClaimType,

ClaimsIdentity.DefaultRoleClaimType);

1. Generate token using claims, current date and other info:

var jwt = new JwtSecurityToken(

issuer: AuthOptions.ISSUER,

audience: AuthOptions.AUDIENCE,

notBefore: now,

claims: identity.Claims,

expires: now.Add(TimeSpan.FromMinutes(AuthOptions.LIFETIME)),

signingCredentials: new SigningCredentials(AuthOptions.GetSymmetricSecurityKey(), SecurityAlgorithms.HmacSha256));

1. Encode and write to response (e.g. to JSON obj field)

var encodedJwt = new JwtSecurityTokenHandler().WriteToken(jwt);

## 4 Token consuming

JWT token can be manually read from request and processed but if there is JwtBearer middleware so you can use:

1. [Authorize], [Authorize(Roles = "admin")] – attribute for controller actions
2. this.User.Identity – to read user info which is restored from JWT token of the request